example to provide the customer with wording selections that already include their event. The remaining variable, guest name of line 303, might be temporarily displayed as "Guest Name Goes Here."

FIG. 3A shows in line 303 that the customer has a choice to make the guest name print in a normal font style like the rest of the invitation or to make it stand out by printing it in bold face style and/or in a selected color. Such typesetting features are easily incorporated in the database associated with wording selections. When it is desired to have such an option, its functionality is stored in an appropriate template.

This invention includes an additional element that is related to individualization. FIG 3A illustrates one example of a template that contains a conditional variable called PLURAL for line 301. This need not be displayed to the customer since the customer need not control this variable. Line 302 is a part of the common text of the template that cannot be changed by the customer. It may need grammatical correction based on the contents of line 301. This correction is required but there is no need to burden the customer with this requirement or to expect that the customer understands that a grammatical correction is required under certain circumstances. An example of the use of this conditional variable is illustrated in FIG. 4.

FIG. 4A is a flow diagram of a routine that implements an example of the processing of a conditional variable, in this case the PLURAL variable discussed in FIG 3A. A template can include conditions, commonly called conditional statements that cause some parts of what is to be printed to change based on the values contained in certain variables. One way such conditions are tested and executed in software code is through the use of "if-then-else" constructs known to software programmers. FIG. 4B shows two examples of the first part of a wedding invitation wording. One example begins with lines 410 and 410; the second example begins with lines 412 and 413. In the first example, line 410 corresponds to the data entry box of FIG 3A, line 301. This is the data entry box for the "Bride's Parents Name" variable, and as the italicized text of FIG 4B shows, it is two people or plural. Line 411 shows the grammatically correct "cordially invite" which should follow. FIG. 4B also shows an example where the "Bride's Parents Name" variable of line 412 is only one person and shows the grammatically correct line 413 "cordially invites" which should follow.

This invention, using such conditional variables in the design of the forms and both the personalization database and the individualization database, allows for conditional variables to affect the indicia that can be displayed or printed from either database. For example, a conditional variable in the personalization database might be used to affect or modify a record in the individualization database prior to display or printing.

Whether for display or printing, FIG 4A, step 401 gets the actual data for testing. In this case the actual data is either FIG 4B, line 410 or line 412. FIG 4A, step 402 tests for a "conditional type variable". If the result of the test in step 402 indicates that the variable is of the conditional type, then step 405 tests for the first in a series of conditional type variables. In this case, the test is a 'compound subject' grammar test. Step 406 shows that any series of similar routines could be included to test for other types of conditional variables.

If the variable tested in step 405 is a "compound subject" variable test, then step 407 tests the value of the current variable. It might do this by searching the variable for the text "and". Finding an "and" means that it is a compound subject meeting the PLURAL test and, in step 409, makes certain that the next printable line of the template is "cordially invite". Or, if the test of step 407 shows that the variable is singular, step 408 substitutes "cordially invites" for the next printable line of the template.

Returning to the test of step 402, if the variable is not a "conditional type variable," step 403 indicates that the current variable has no conditions so use the actual data variable. In practice, other types of tests could be made here. One example would be an attributes test for bold face type. Step 404 formats the variable for the output device, a printer or display device. Formatting in this example would include setting the font, font size, color, and print coordinates for the line of text.

FIG. 5 illustrates an individualization data entry mechanism in one embodiment. When collecting information from a user, a web page is generated on the server system and sent to the client system. The layout of the web page is designed to facilitate data entry by the user in an easily understood manner. FIG. 5 illustrates an outline format of a sample form to be filled in. The sample form contains various sections identified by letters A, B, and C. In particular, the illustrated example shows a form for collecting names and addresses such as might be used for a

guest list. Section A shows data entry boxes for name and address data along with an example of how the customer should enter this data.

Section B of FIG 5 permits the customer to assign certain attributes associated with the data. In the depicted case, the three text lines are made part of 'radio buttons' in common use in the design of web pages. Radio buttons, like the push buttons on car radios permit selection of only one item of a group of items. This is well known to both software programmers and web page designers. In the section B example, only one of the three options is possible for a given guest. When this form is present, the 'Invite This Guest' radio button might be the default condition.

The attributes shown as 'radio buttons' in FIG 5, section B might be used to differentiate the printing of invitation envelopes versus announcement envelopes. Or, as in the case of wedding invitations, two different templates might be used. One template contains an invitation wording; another, an announcement wording. In this case, should the customer press select the "Send Announcements Only" option for at least one guest, logic built into the template would advise the customer that he should also select an additional template worded specifically for announcements.

Section C of FIG 5 shows form buttons that control access to a particular record. Specifically, this example form is used to collect guest names for the personalized printing of that guest name on each wedding invitation. In this case, the section A "first name" and "last name" variables are merged together with a space separator and are substituted as variable in the invitation wording. However, this same data along with the address are used when printing envelopes. Separating the first and last names permits logic associated with the database storing guest name and address data to sort the guest list based on the last name. This might be useful if the customer wishes to have an alphabetically ordered printout of the guest list on the printer attached to the client system. The printed guest list sent to the client system is not a printed product of this invention but simply a customer report printed on ordinary paper.

The buttons shown in Section C are used to select the record for data entry or editing. In this case, the customer has the option of pressing the NEXT or PREVIOUS button to retrieve and display the appropriate adjacent record within and retrieved from the individualization database. When so pressed, the currently displayed data on the client system is sent to the server